cse@buffalo

Fingerprint Research

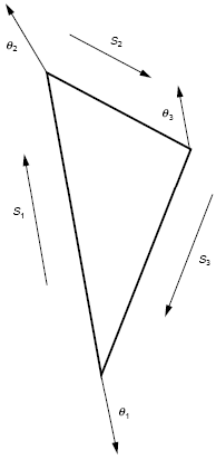
Center for Unified Biometrics and Sensors,
University at Buffalo

Overview of Research Topics

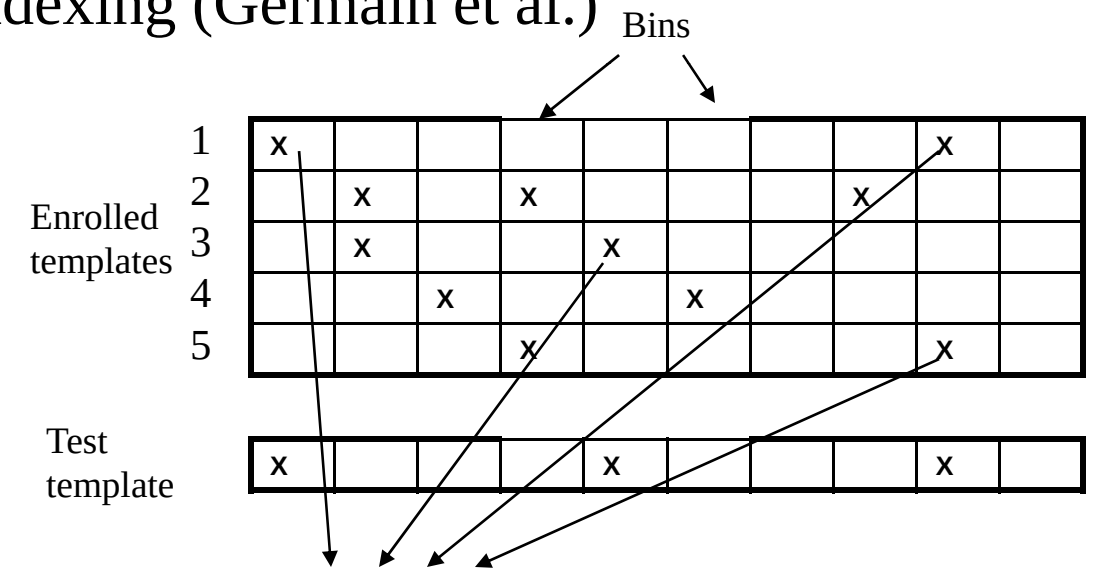
- Image Preprocessing, Segmentation, Binarization
  - -STFT based
  - -Directional Median Filtering with Adaptive (quality based) Window Size & Harris corner point based segmentation
- Feature (minutia extraction)
  - - Contour based; searching for turns in contours
- Minutia Matching
  - - Triplet and secondary feature matching
- Verifying Minutia Matching by Correlation Methods
- Fingerprint Indexing
  - - Tree based; branches correspond to local minutia geometry
- Cancelable Fingerprint Templates
  - - Hashes are symmetric functions of local neighborhood minutiae

Previous Research in Indexing

- Classification into predefined classes (arch, loop, whorl)
- Continuous Classification or Filtering (e.g. Fingercode)
- Triplet based indexing (Germain et al.)



Each minutia triplet is associated with some bin

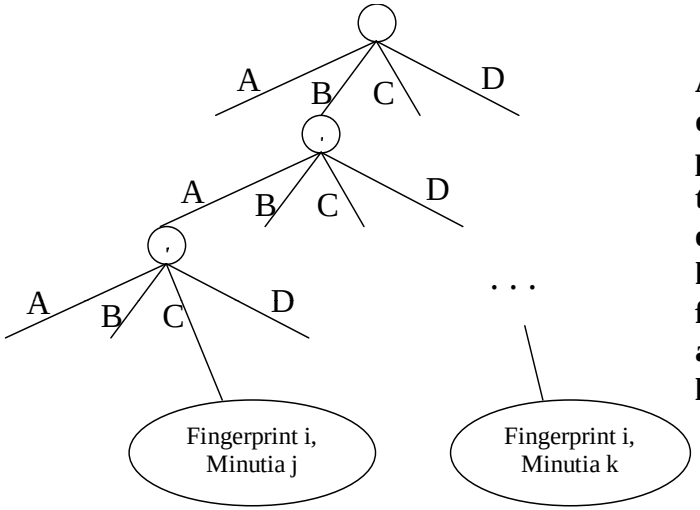


Retrieved indexes: 1, 3, 1, 5

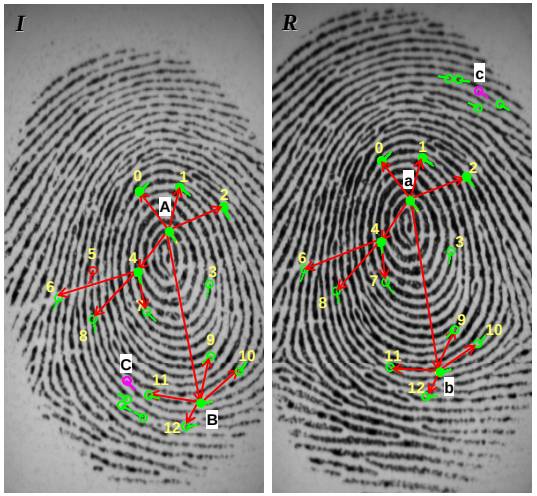
Count fingerprints with most matches: 1(2 times), 3 (1 time), 5 (1 time)

Indexing Idea

Index structure for fingerprints:



A,B,C,D denote quantization bins of k-plets. Going down the tree is equivalent to the expansion of neighboring k-plet. Note that each fingerprint could be associated with multiple leaf nodes.

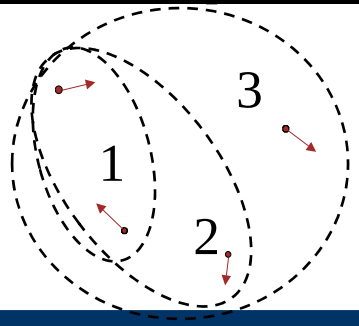


Coupled breadth-first fingerprint matching: expansion of k-plets is similar to tree searching.

Algorithm Features:

- True tree-based indexing
- Local connections between triplets (bins) are accounted for
- Matching test fingerprint against the tree is equivalent to matching against one template
- Fewer bins, but more structure and separability

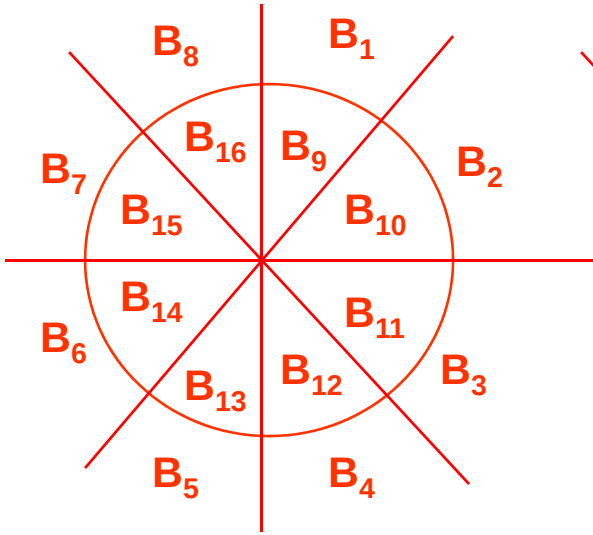
Current implementation design:

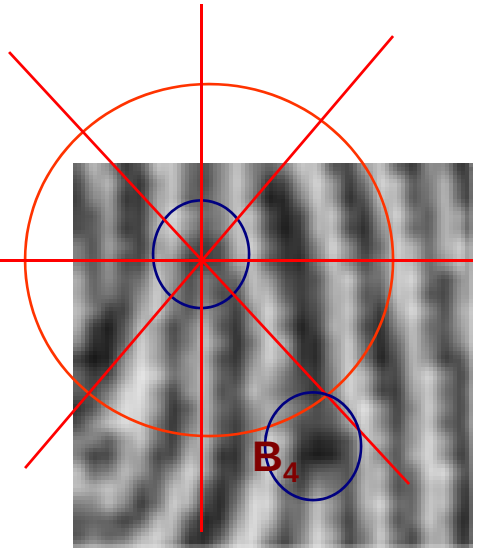


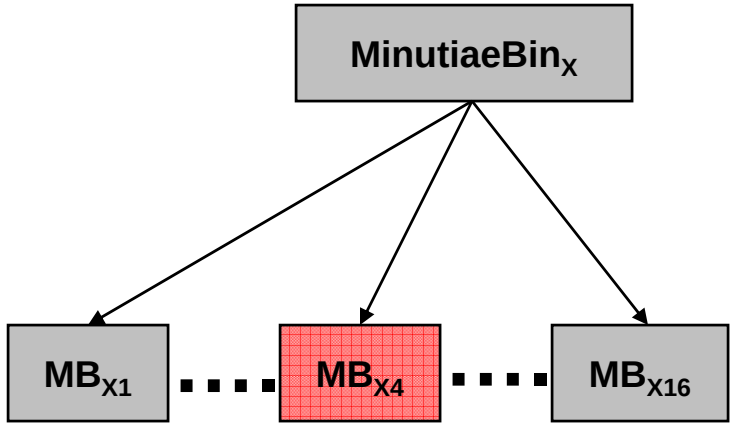
- **K-plet is a single minutia**
- **The expansion of k-plets = search for nearest minutia**

Branch Selection

- Branching on each level is done based the relative features of the current minutiae and its nearest neighbor.
- Finite number of bins are used to handle continuous-valued features.





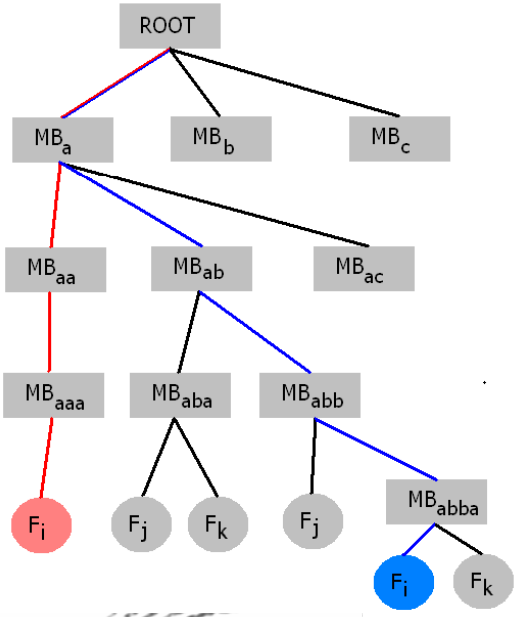


16-bins based on minutia point position relative to centre point.

Fingerprint Enrollment

- Fingerprint preprocessing and minutiae point extraction.
- One minutia point is selected as the root
- For each neighboring minutiae point, we traverse down the tree one level at a time and add the fingerprint at the appropriate leaf node.
- The process can be repeated for different points.

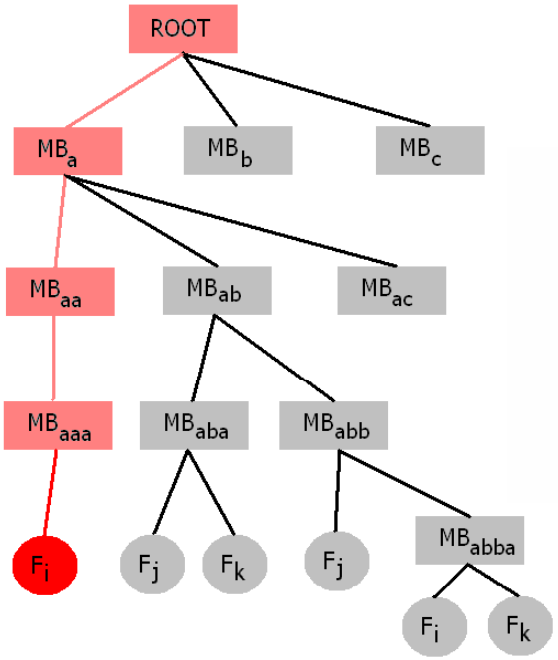
Thus we see that we do not need to rebuild the tree at a later stage while enrolling additional users into the system.



Fingerprint Matching

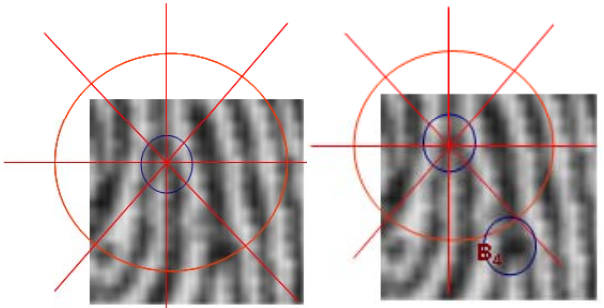
Single path matching:

- Fingerprint preprocessing and minutiae extraction.
- Select one point as root and find nearest neighbor. Calculate features of this point (neighbor) w.r.t. the current minutia.
- Based on the feature values, traverse down the tree, taking one minutia point at a time.
- Repeat by taking other minutiae as roots



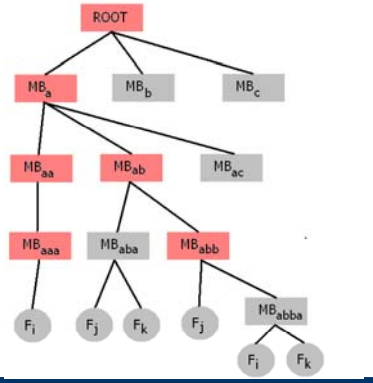
Multiple path matching:

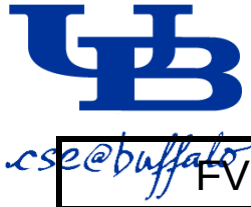
- If some minutia is close to the bin boundary, explore alternative matching paths



Searching Phase

Enrollment Phase





Single path search:

FVC 2002 DB1	N = 5	N = 6	N = 7	N = 8
Correct	0.43	0.41	0.19	0.10
No Matches Found	0.08	0.55	0.81	0.90
Incorrect	0.49	0.04	0	0
Average Returned Matches	1.42	1.03	1.00	1.00

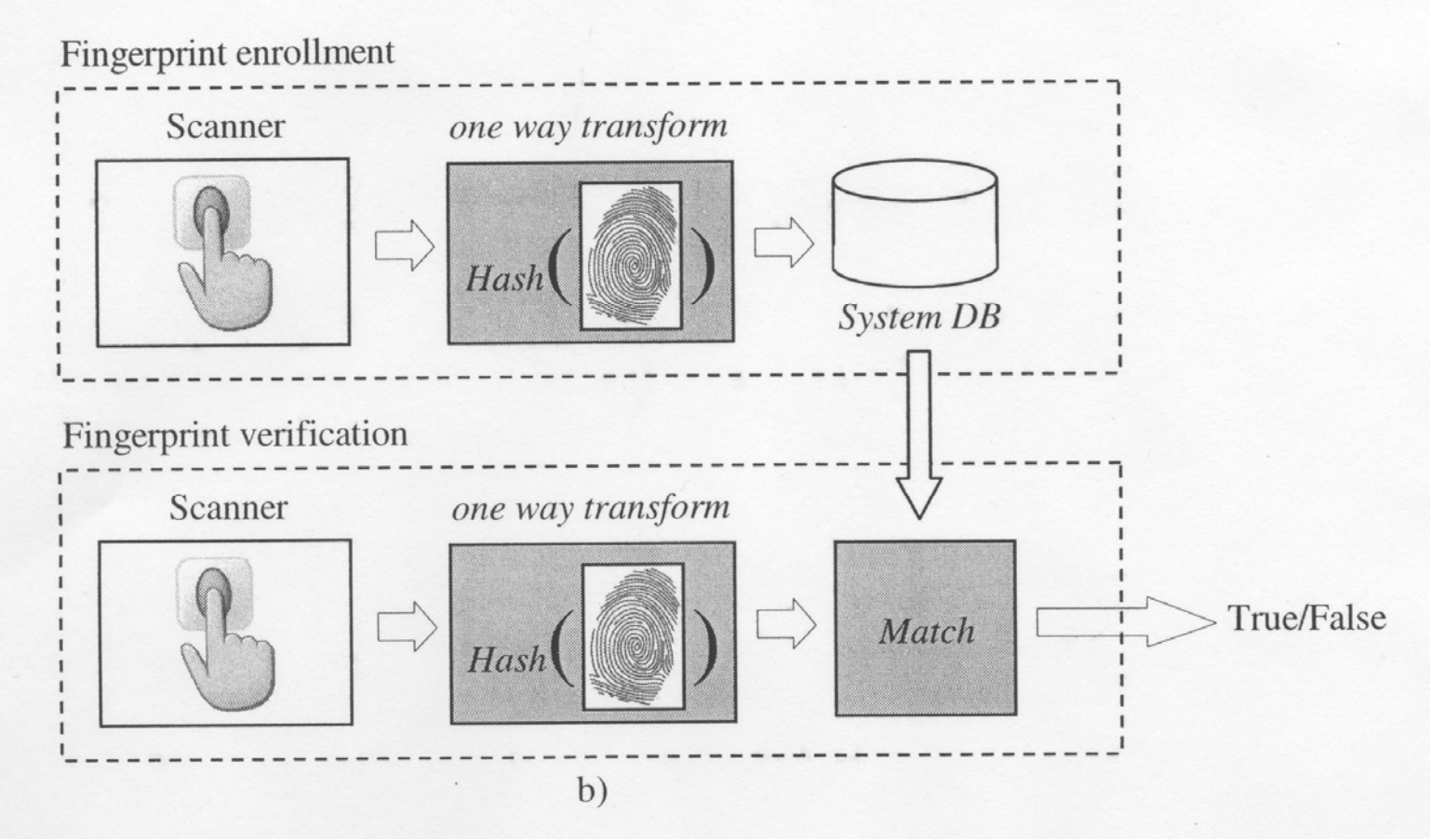
Multiple path search:

FVC 2002 DB1	N = 5	N = 6	N = 7	N = 8
Correct	0.32	0.58	0.44	0.30
No Matches Found	0.02	0.28	0.55	0.70
Incorrect	0.66	0.14	0.01	0
Average Returned Matches	1.29	1.08	1	1

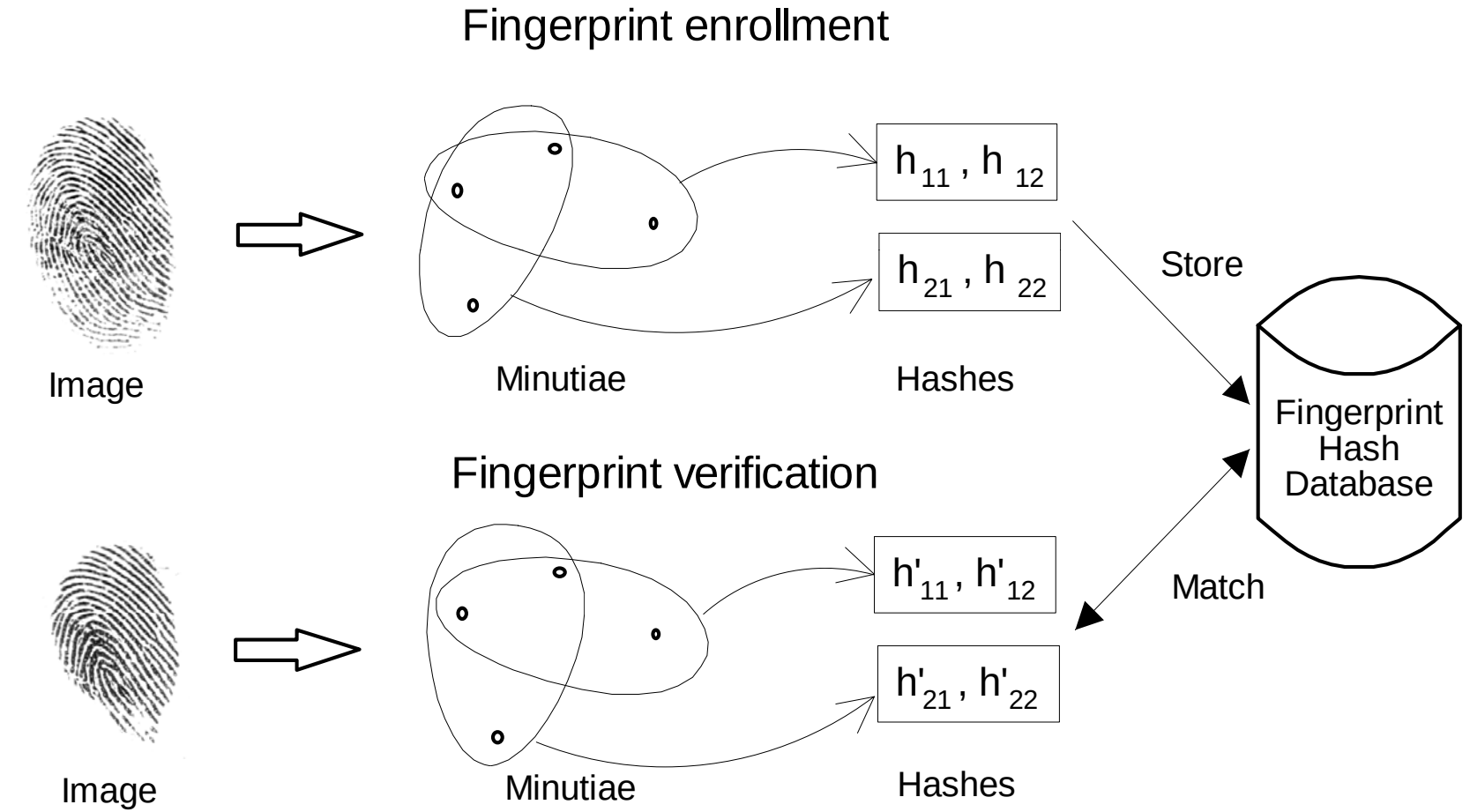
Time:

FVC 2002 DB1	N = 5	N = 6	N = 7	N = 8
Single Path Search	0.112	0.032	0.028	0.028
Multiple Path Search	0.106	0.096	0.146	0.230

Cancelable Fingerprint Templates

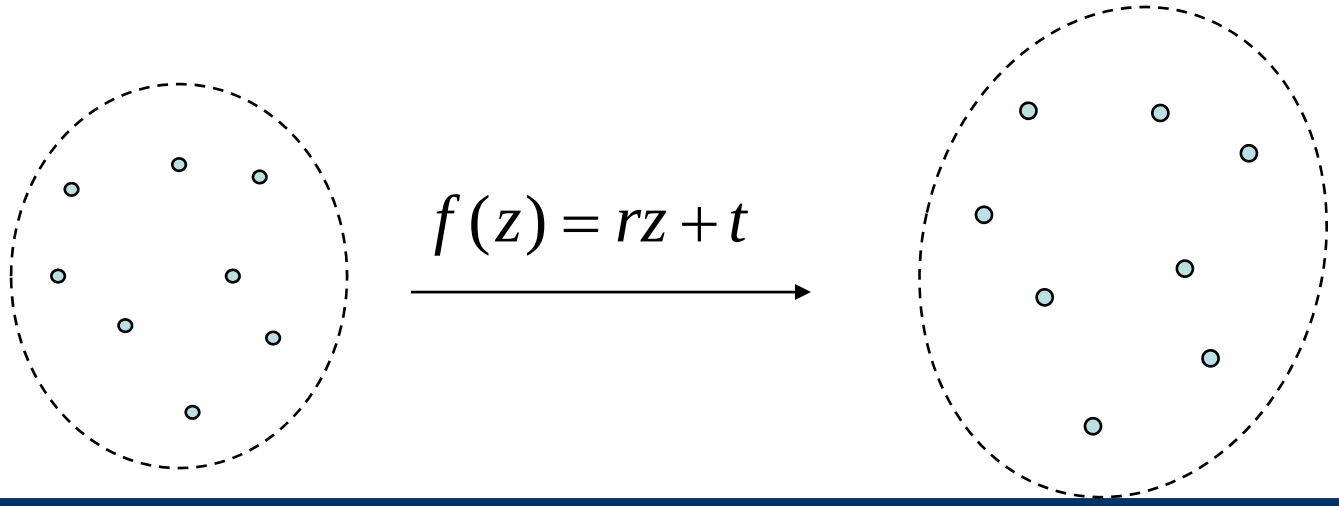


Local neighborhood minutia for hashes:



Assume rigid transformation between fingerprints:

If $(c_1, c_2, \ldots, c_n)$ is a set of minutia points of first fingerprint and $(c_1', c_2', \ldots, c_n')$ is a set of minutia points of second fingerprint (same finger), then we assume that there is a transformation $f(z) = rz + t$ such that $c_i' = f(c_i) = rc_i + t$ for any $i = 1, \ldots n$.



Hash Functions of Each Local Minutia Set:

Consider following functions of minutia positions:

$$h_1 = h_1(c_1, c_2, \ldots, c_n) = c_1 + c_2 + \ldots + c_n$$

$$h_2 = h_2(c_1, c_2, \ldots, c_n) = c_1^{\,2} + c_2^{\,2} + \ldots + c_n^{\,2}$$

$$\vdots$$

$$h_m = h_m(c_1, c_2, \ldots, c_n) = c_1^{\,m} + c_2^{\,m} + \ldots + c_n^{\,m}$$

The values of these symmetric functions do not depend on the order of minutia points.

Hash Functions of Transformed (Test) Fingerprint:

$$h_1' = h_1(c_1', c_2', \ldots, c_n') = c_1' + c_2' + \ldots + c_n'$$
$$= (rc_1 + t) + (rc_2 + t) + \ldots + (rc_n + t)$$
$$= r(c_1 + c_2 + \ldots + c_n) + nt = rh_1(c_1, c_2, \ldots, c_n) + nt$$

$$h_2' = h_2(c_1', c_2', \ldots, c_n') = c_1'^2 + c_2'^2 + \ldots + c_n'^2$$
$$= (rc_1 + t)^2 + (rc_2 + t)^2 + \ldots + (rc_n + t)^2$$
$$= r^2(c_1^2 + c_2^2 + \ldots + c_n^2) + 2rt(c_1 + c_2 + \ldots + c_n) + nt^2$$
$$= r^2 h_2(c_1, c_2, \ldots, c_n) + 2rth_1(c_1, c_2, \ldots, c_n) + nt^2$$

Finding Transformation Parameters From Hash Values:

$$h_1' = rh_1 + nt$$
$$h_2' = r^2 h_2 + 2rth_1 + nt^2$$

And r,t can be calculated given h_1, h_2, h_1', h_2'

In practice:

- Consider also hashes of minutia directions

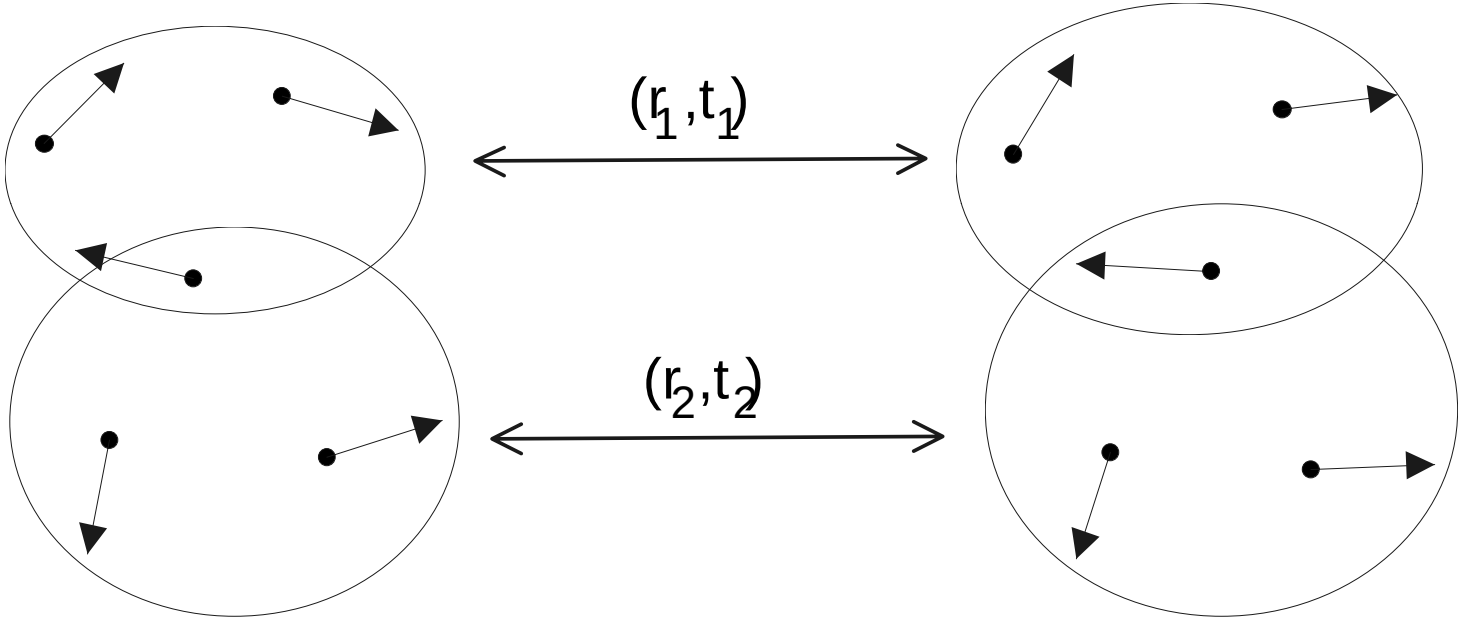
$$g_i(d_1, d_2, \ldots, d_n) = d_1^{\,i} + d_2^{\,i} + \ldots + d_n^{\,i}$$

- Search for transformation parameters minimizing distance functions

$$dist\ (h_i, h_i', g_i, g_i', r, t) = \alpha_1 \mid h_1' - rh_1 - nt \mid + \alpha_2 \mid h_2' - r^2 h_2 - 2rth_1 - nt^2 \mid$$
$$+ \alpha_3 \mid g_1' - rg_1 - nt \mid + \alpha_4 \mid g_2' - r^2 g_2 - 2rtg_1 - nt^2 \mid$$

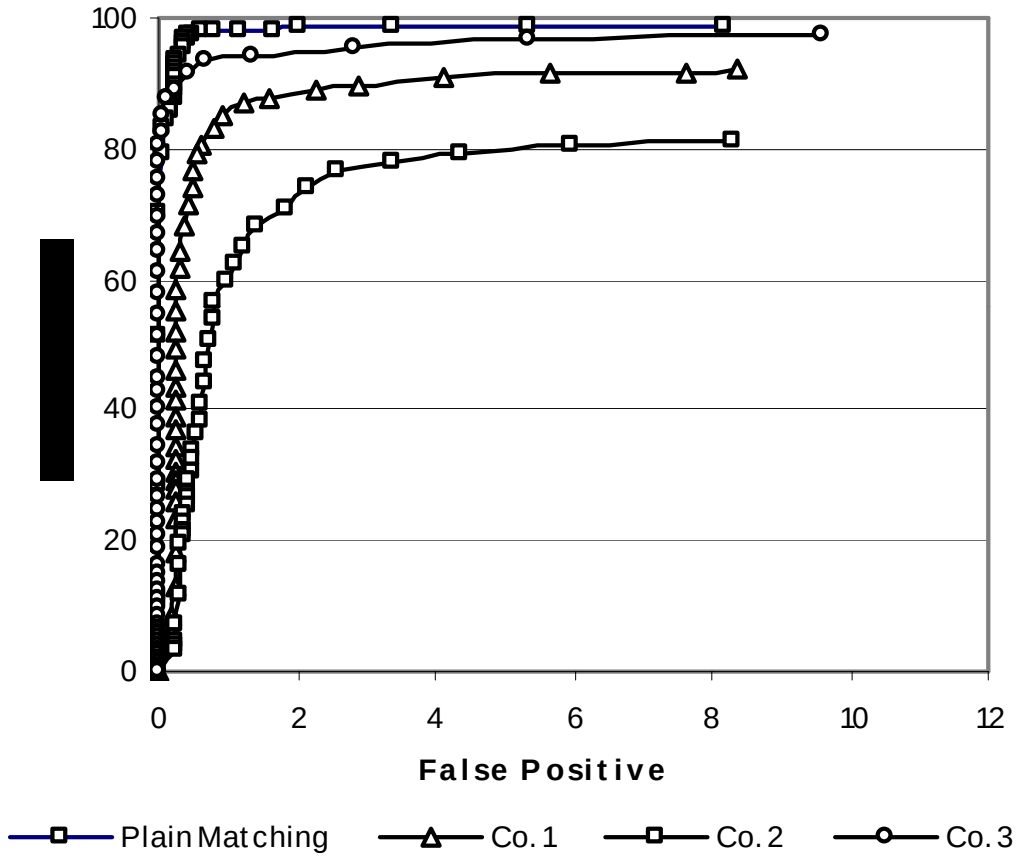
Global Matching – Combine Local Matches

- Finding global transform parameters:



Find a cluster of $(r_i, t_i), \quad i = 1, \ldots, n$

Experimental Results



Co.3: 3 minutiae and 2 hashes
ERR = 3%
Original no-hash matching
ERR = 1.7%

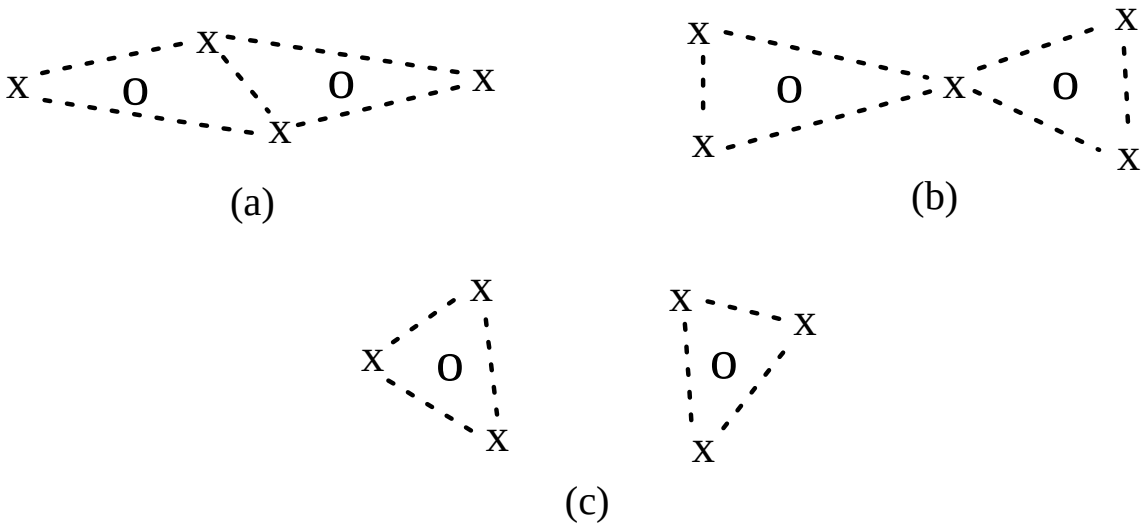
Co.2: 3 minutiae and 1 hash
Co.1: 2 minutiae and 1 hash

tested on FVC2002 set,
with 2800 genuine tests
and 4950 impostor tests

Security

- If the number of stored hash functions h_i is less than the number of minutia points, it is not possible to find the positions of minutia points c_j from local hash values.

- Using system of hash equations is difficult, since it is not known which minutia correspond to particular hash value.



Generating New Hashes

$s_i = s_i(c_1, c_2, \ldots, c_n), i = 1, \ldots, m$ - basis of symmetric polynomial functions of degree less or equal than m.

Then $s_i' = s_i(rc_1 + t, rc_2 + t, \ldots, rc_n + t)$ are also symmetric, polynomial functions of degree less or equal than m.

⇒ For some predetermined F : $s_i' = F(r, t, s_1, \ldots, s_m)$

⇒ r, t can be found from hash sets $\{s_1, \ldots, s_m\}$ and $\{s_1', \ldots, s_m'\}$

Conclusion: *any basis in a set of symmetric polynomial functions can be used for hashing*

Example:

$$s_1(c_1, c_2, \ldots, c_n) = a_{11} h_1(c_1, c_2, \ldots, c_n)$$

$$s_2(c_1, c_2, \ldots, c_n) = a_{21} h_2(c_1, c_2, \ldots, c_n) + a_{22} h_1^2(c_1, c_2, \ldots, c_n) + a_{23} h_1(c_1, c_2, \ldots, c_n)$$